CIRSSE-R-64

NASA-CR-192747

[illegible]

TECHNICAL REPORTS

[illegible]

N 61-CR

(NASA-CR-192747) SPHERICAL-OBJECT REPRESENTATION AND FAST DISTANCE COMPUTATION FOR ROBOTIC APPLICATIONS (Rensselaer Polytechnic Inst.) 16 p

N93-71643

153775

Unclas

P. 16

Z9/61 0153775

CIRSSE

Center for Intelligent Robotic Systems for Space Exploration

Rensselaer Polytechnic Institute
Troy, New York 12180-3590

SPHERICAL-OBJECT REPRESENTATION AND FAST DISTANCE COMPUTATION FOR ROBOTIC APPLICATIONS

by

Josep Tornero

Rensselaer Polytechnic Institute
Electrical, Computer, and Systems Engineering
Troy, New York 12180-3590

September, 1990

CIRSSE REPORT #64

SPHERICAL-OBJECT REPRESENTATION AND FAST DISTANCE COMPUTATION FOR ROBOTIC APPLICATIONS

Josep Tornero[1]

Departamento de Ingenierías de Sistemas,
Computadores y Automática (DISCA)
Universidad Politécnica de Valencia, P.O.B. 22012
E-46071 Valencia. SPAIN.

Gregory J. Hamlin and Robert B. Kelley

Center for Intelligent Robotics Systems
for Space Exploration (CIRSSE)
Rensselaer Polytechnic Institute, 8015 CII
Troy, NY 12180, USA.

Abstract

A three-dimensional object representation technique for generating spherical-geometries and a fast procedure for computing distances using this geometry is presented. An object is approximated by an infinite number of spheres. The shortest distance between two objects is obtained by finding the two spheres, one from each object, that are closest. Exceptional numerical results have been obtained, for example, the maximum time for computing self-collision for a standard PUMA robot-arm is equal to 2.30 milliseconds with an error in distance of less than 1cm. This makes the new technique an invaluable tool for computing distances and therefore permits collision-detection in real-time. This technique has been applied to a complex robotic system consisting of two PUMA robots, each mounted on a three-degree of freedom platform used at the CIRSSE to study robotic assembly of structures in space.

Introduction

The fast computation of distances is an important part of many robotic applications. Distances between the elements of a robot manipulator, between elements of cooperating robot arms and between the robotic elements and objects in the environment may all need to be computed to guarantee collision free payload transportation and manipulation. Fast distance computation is an asset to trajectory planning and manipulation planning and is essential for the on-time implementation of collision detection and obstacle avoidance strategies. In the context of vehicular navigation in both 2-D and 3-D environments, fast distance computations are also important. Potential fields and other similar functions find use in trajectory planning and can similarly benefit from fast distance computation. Although all of these applications are not explicitly addressed in this paper, it is clear that the approach presented here is appropriate whenever distance computations have to be fast.

[1]Professor Tornero is Associate Professor at the Universidad Politécnica de Valencia (Spain) and at present Visiting Researcher at CIRSSE

This paper presents a modeling technique which is conceptually between the bubble model of O'Rourke and Badler [1] and the generalized cylinders of Agin [2]. The model is derived from the basic idea that an object can be approximated by an infinite number of spheres. Hence it retains the advantages of the bubble model and avoids the complexity of the generalized cylinder. Using spheres, the shortest distance between two objects can be obtained by finding the two spheres, one from each object, that are closest. Given these spheres, the computation of the shortest distance is trivial. Further, the intrinsic symmetry of the spheres eliminates the need to consider orientation.

Before presenting the sphere-based object representation technique and the associated fast distance computational procedure, a brief discussion of the problem and the work of others is given.

Path planning is based on having a good description of the world with its constraints as well as a clear description of the required motion. A popular method of planning gross motions has been to create a configuration space (C-space) model [3, 4]. For a moving point, the configuration space obstacles are easy to compute. The VGraph algorithm has been applied successfully in 2-D and with cartesian manipulators [5]. A recursive version of the VGraph algorithm has been developed to be used in 3-D [6, 7]. The extension of this technique to articulated manipulators has been found to be very computationally expensive.

Several attempts have been made to describe the space free of obstacles (free-space) rather than the space of obstacles. Along this line, an early idea given by Brooks [8] was to define conical volumes between objects. Later on Lozano-Perez and Brooks [9] presented the idea of obstacles included in rectangular cell which were subdivided recursively into smaller ones. This technique requires a great amount of computation for robot arms.

Some authors use spatial occupancy enumeration to model the freespace, and then determine a collision free path using explicit spatial planning. Octrees, a special form of spatial occupancy enumeration employs a successive subdivision of space into octants. Cells with different sizes have been used to reduce the number of collision-detection computations [10]. Faverjon [11] describes an algorithm for transforming cartesian obstacles into obstacles in the space of the first three joints of a manipulator. This is based on a hierarchical structure. The major difficulty in any of these schemes, in addition to the price of initial computation, is the large computational effort required to modify the configuration space or octree representation when objects move and especially when they change orientation.

Since many elements of robotic systems are bounded by plane surfaces, polyhedra have been used to develop procedures for computing distances [12]. However, the large number of comparisons required for such elements is highly time consuming. Thus, some authors have recast the problem into a standard linear programming form. Sometimes, norm 1 and norm ∞ rather than Euclidean distances are used to save computational effort [13]. Objects of revolution have been used to obtain faster algorithms. However, this objects have the disadvantage of combining curvilinear side surfaces with plane end surfaces. Replacing the end planes with hemispheres is one way to speed up the algorithm [14].

Several optimal-control techniques are based on a good description of distances between objects. The problems are defined analytically: however, such problems are generally solved numerically [15] and therefore distances can be computed at each step. Distances are also required in order to create artificial potential fields. Repulsive forces are obtained as functions of the distances to the other objects [16]. Some papers are interested in collision between mobile objects [17, 18, 19]; however, the algorithms available for static objects do not work efficiently with moving objects.

The thrust of this paper is to first describe an object representation technique based on spherical-volumes and second to develop a fast procedure for computing distances between these volumes. The good numerical results obtained validate thoroughly the theory as well as makes this technique a valuable tool for distance computation and collision detection in real-time. An existing robot system has been used to show how this technique may be used for complex systems.

Object Representation

The modeling technique is based on the idea that any real object can be approximated by an infinite number of spheres. For a particular object many different sets of spheres can be generated, depending of the degree of accuracy required. In general accuracy is directly related to the complexity of the process for obtaining the set of spheres. Later on, we will introduce the idea of degree-of-freedom as a first measurement of this complexity.

The way of generating a set of spheres for a given object is provided by introducing the concept of dynamic-spheres. A dynamic-sphere is defined as a sphere whose center **P(x,y,z)** can move in a three-dimensional space and whose radius is a function of the position at each moment **R(x,y,z)**.

The object model corresponds to the volume swept by the dynamic-sphere when moving in a bounded subspace. The subspace can be defined by

1. A set of inequality constraints,e.g., $x_0 \leq x \leq x_1$.
2. A set of functional constraints in the form $f_i(x, y, z) = 0$.

The number of functional constraints will determine the degrees of freedom for the center of the dynamic-sphere. For example, a two-degree of freedom geometry is obtained when one functional constraint is introduced. In parametric representation, that is

$$f_1(x, y, z) = 0$$

$$P = P(\lambda_1, \lambda_2), \quad R = R(\lambda_1, \lambda_2) \tag{1}$$

The complexity of the volume swept by the dynamic-sphere depends on the functions $\mathbf{P}(\lambda_i)$ and $\mathbf{R}(\lambda_i)$, as well as the range of the $\lambda's$. To make the problem as standard as possible, λ_i will be fixed, $\lambda_i \in [0, 1]$, $\forall i$.

The simplest class of objects for the one-degree of freedom geometry is obtained by considering linear functions of the form,

$$\begin{cases} P = P_0 + \lambda P_1 \\ R = R_0 + \lambda R_1 \end{cases} \tag{2}$$

We define a spherical-cone as the volume swept by a dynamic-sphere whose center and radius obey linear functions in the parameter λ as shown in equations 2. Particular cases of this geometry for λ varying within a given range. such as spherical-cylinders, etc., can be seen in Figure 1.

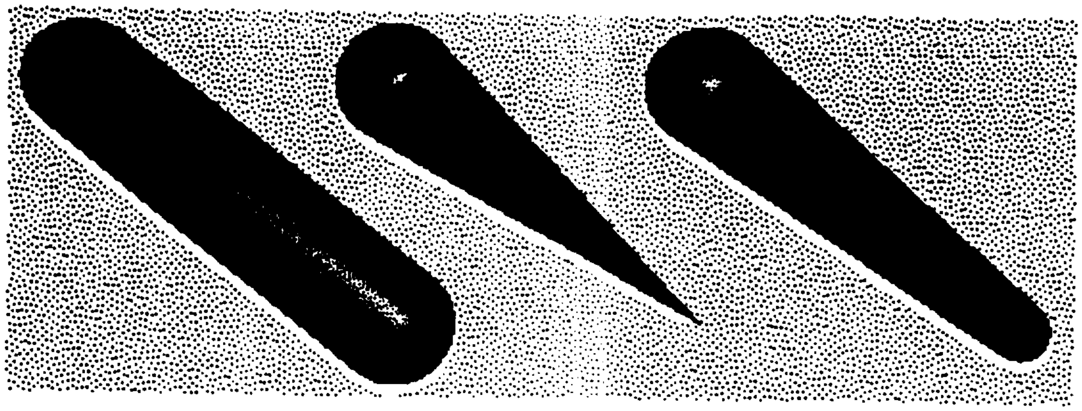

Figure 1: Spherical-cones

For a spherical-cone i, Equations 2 can be rewritten in terms of the radius and center of the end-spheres at the two extremes of the volume, $(\mathbf{P}_{i0}, \mathbf{R}_{i0})$ and $(\mathbf{P}_{i1}, \mathbf{R}_{i1})$, in the following way,

$$\left\{ \begin{array}{l} P_i = P_{i0} + \lambda_i \ (P_{i1} - P_{i0}) \\ R_i = R_{i0} + \lambda_i \ (R_{i1} - R_{i0}) \end{array} \right. \tag{3}$$
$$\lambda_i \in [0,1]$$

The degree and the angle of convergence of a spherical-cone can be defined respectively by

$$\eta_i = \frac{(R_{i1} - R_{i0})}{\mid P_{i1} - P_{i0} \mid}, \quad \alpha_i = \arcsin \eta_i \tag{4}$$

For values of degree of convergence equal or greater than 1, $(\eta_i \geq 1)$, the volumes degenerate into a simple sphere equal to the largest end-sphere.

For the two-degree of freedom case, the simplest geometry which can be represented is a dynamic-sphere moving tangentially between two planes, which is called as a spherical-plane. In a similar way as a spherical-cone is bounded by two end-spheres, a spherical-plane is bounded by four end-spherical-cones which constitute the sides of the plane. A volume i of this kind can be described by two spherical-cones with a common end-sphere, $[(\mathbf{P}_{i0}, \mathbf{R}_{i0}), (\mathbf{P}_{i1}, \mathbf{R}_{i1})]$ and $[(\mathbf{P}_{i0}, \mathbf{R}_{i0}), (\mathbf{P}_{i2}, \mathbf{R}_{i2})]$, which constitute two continuous sides. The volume is then generated from a dynamic-sphere which follows the following equations

$$\left\{ \begin{array}{l} P_i = P_{i0} + \lambda_{i1}(P_{i1} - P_{i0}) + \lambda_{i2}(P_{i2} - P_{i0}) \\ R_i = R_{i0} + \lambda_{i1}(R_{i1} - R_{i0}) + \lambda_{i2}(R_{i2} - R_{i0}) \end{array} \right. \tag{5}$$
$$\lambda_{i1} \& \lambda_{i2} \in [0,1]$$

According to Equations 5, the center of the dynamic-sphere moves inside a parallelogram, which can be easily cut-off by lines, defined as inequality constraints, in the form of linear combination of parameters $\lambda's$,

$$a_1 \cdot \lambda_1 + a_2 \cdot \lambda_2 \leq 1 \tag{6}$$

A more general, but still simple, two degree of freedom structure can be obtained from two opposite end-spherical-cones of the plane, $[(\mathbf{P}_{i0}, \mathbf{R}_{i0}), (\mathbf{P}_{i1}, \mathbf{R}_{i1})]$ and $[(\mathbf{P}_{i2}, \mathbf{R}_{i2}), (\mathbf{P}_{i3}, \mathbf{R}_{i3})]$. In this case, the geometrical locus of the center of the dynamic-sphere is not forced to be a

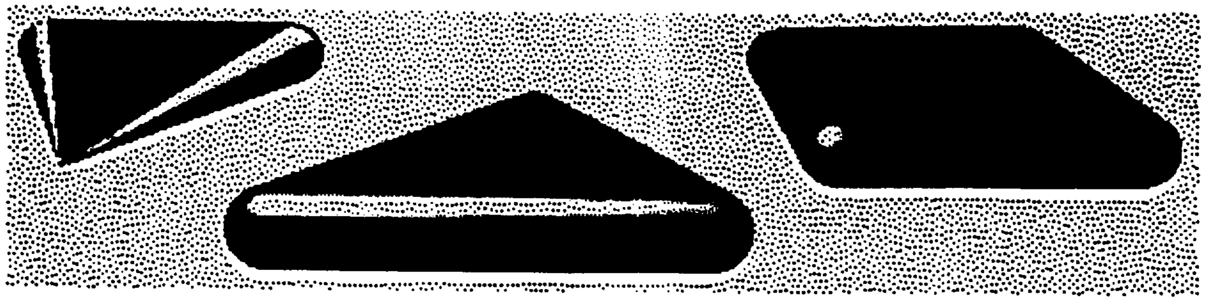

Figure 2: Spherical-planes

parallelogram. However, several relations between the two spherical-cones must be stated in order belong to the same spherical-plane.

Several kinds of spherical-planes can be obtained with this two-degree of freedom geometry as depicted in figure 2. Spherical planes are bounded at least by three spherical-edges which are in the class of spherical-cones or one-degree of freedom geometries. These spherical-edges intersect in spherical-vertices which corresponds to the zero-degree of freedom geometries.

Dynamic-spheres with three degrees of freedom are not considered in this paper, given that the most outstanding advantage of this object representation technique comes from the possibility of generating three-dimensional models with less than three degrees of freedom. However, the basic idea can be easily extended to dynamic-spheres moving in three or even more degrees of freedom. Extra degrees of freedom can be introduced in order to increase the accuracy of the representation or to consider aspects such as the time-factor for describing movements of objects in the space.

Procedure for Fast Distance Computation

The problem of finding the shortest distance between two bodies whose volumes have been generated by the movement of two dynamic-spheres. (P_i, R_i) and (P_j, R_j), along their trajectories can be stated as a Min-Max problem as follows,

$$\min_{\lambda_i, \lambda_j}\{f\{\mid P_i(\lambda_i) - P_j(\lambda_j) \mid - R_i(\lambda_i) - R_j(\lambda_j)\}\} \tag{7}$$

where $f\{x\} = x$ if $x \geq 0$
$= 0$ otherwise

The problem can be expressed in terms of finding two spheres, each belonging to a distinct geometric structure, with the shortest distance between them.

Distance Between Volumes Generated from Dynamic Spheres with One-degree of Freedom

The kind of functions selected in the definition of the movement of centers $P_i(\lambda_i)$ and evolution of the radius $R_i(\lambda_i)$ affects directly the complexity of the Min-Max problem. Even for the

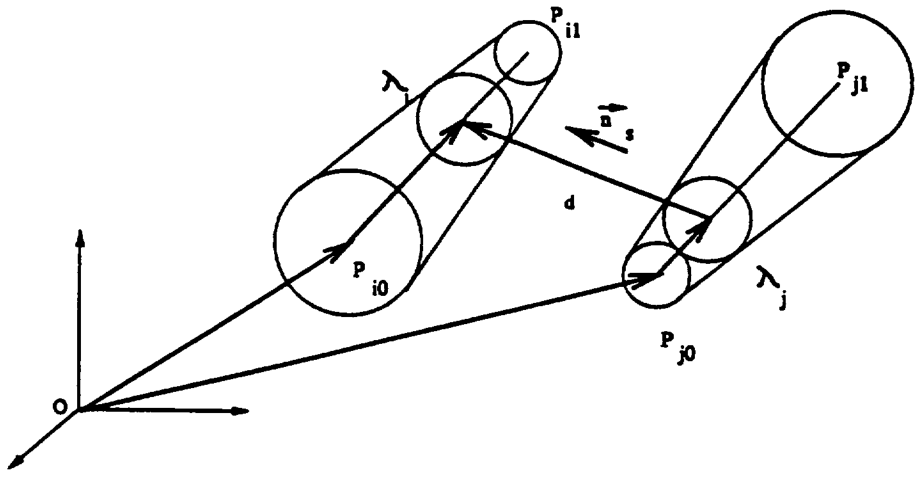

Figure 3: Distance between two spherical-cones

simplest objects generated from spheres with one degree of freedom and linear functions the use of Euclidean distances means that, the set of resulting equations involve square roots.

The set of non-linear equations for computing the distance between two spherical-cones as shown in Figure 3 can be simplified resulting in two sets of vector equations which solve the problem in two steps.

1. Compute the direction of the shortest distance.

$$\begin{aligned} \vec{n}_s \cdot (P_{i1} - P_{i0}) &= R_{i1} - R_{i0} \\ \vec{n}_s \cdot (P_{j1} - P_{j0}) &= -(R_{j1} - R_{j0}) \end{aligned} \tag{8}$$

   where the unknown $\vec{n}_s$ is the unitary vector in the direction of the shortest distance between volumes and which is perpendicular to both surfaces.

2. Compute the spheres involved.

$$P_{i0} + \lambda_i \cdot (P_{i1} - P_{i0}) = P_{j0} + \lambda_j \cdot (P_{j1} - P_{j0}) + d \cdot \vec{n}_s \tag{9}$$

   where λ_i and λ_j are the parameter values which define the spheres involved
   d is the distance between the centers of these spheres

The distance is obtained as follows.

$$Distance = d - R_i(\lambda_i) - R_j(\lambda_j) \tag{10}$$

The unitary vector $\vec{n}_s$ perpendicular to the surfaces can be computed by using a pseudo-polar coordinate system representation, as shown in Figure 4, and is based on the three following vectors,

$$\begin{aligned} \vec{v}_i &= \frac{(P_{i1} - P_{i0})}{\mid P_{i1} - P_{i0} \mid} \\ \vec{v}_j &= \frac{(P_{j1} - P_{j0})}{\mid P_{j1} - P_{j0} \mid} \end{aligned} \tag{11}$$

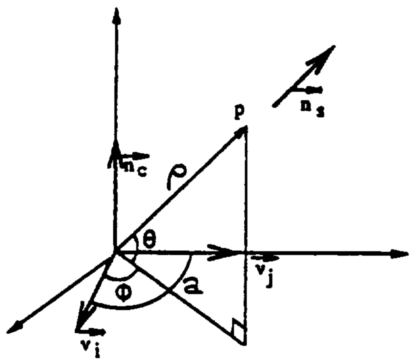

Figure 4: Pseudo-Polar Coordinate System

$$\vec{n}_c = \vec{v}_i \times \vec{v}_j$$

Any point in space, or any vector centered at the coordinate origin can be described in terms of a linear combination of these three unitary vectors. Thus

$$\vec{n}_s = g_{v_i} \cdot \vec{v}_i + g_{v_j} \cdot \vec{v}_j + g_{n_c} \cdot \vec{n}_c \tag{12}$$

where $g_{v_i} = \rho \cdot \cos\theta \cdot \frac{\sin(a-\phi)}{\sin a}$
$g_{v_j} = \rho \cdot \cos\theta \cdot \frac{\sin\phi}{\sin a}$
$g_{n_c} = \rho \cdot \sin\theta$
$a = \arccos(\vec{v}_i \cdot \vec{v}_j)$

For the coordinate system introduced and using the degrees of convergence, the inner-products in Equations 8 can be rewritten as

$$\begin{aligned} g_{v_i} + g_{v_j} \cdot \cos a &= \eta_i \\ g_{v_i} \cdot \cos a + g_{v_j} &= -\eta_j \end{aligned} \tag{13}$$

By solving these equations, we get

$$\tan\phi = \frac{\eta_j + \eta_i \cdot \cos a}{\eta_i \cdot \sin a}, \quad \cos\theta = \frac{\eta_i}{\cos\phi}, \quad \rho = 1 \tag{14}$$

In the special case of two cylinders, the value of vector $\vec{n}_s$ is obtained directly as $\vec{n}_s = \vec{n}_c$.

When the vector $\vec{n}_s$ has been computed, the values of the parameters $\lambda's$ can be obtained by solving the set of scalar linear equations or directly from the vector equation 9. Multiply this equation by $\vec{n}_{s_j} = \vec{n}_s \times \vec{v}_j$ and $\vec{n}_{s_i} = \vec{n}_s \times \vec{v}_i$ respectively, to obtain λ_i and λ_j.

$$\lambda_i = \frac{(P_{j0} - P_{i0}) \cdot \vec{n}_{s_j}}{(P_{i1} - P_{i0}) \cdot \vec{n}_{s_j}}$$

$$\lambda_j = -\frac{(P_{j0} - P_{i0}) \cdot \vec{n}_{s_i}}{(P_{j1} - P_{j0}) \cdot \vec{n}_{s_i}} \tag{15}$$

When at least one of the $\lambda's$ is out of range ($e.g. \lambda_i \notin [0,1]$), the distance between the corresponding end-sphere ($P_i^* = P_i(0)$ or $P_i^* = P_i(1)$) and the other volume is computed. The problem is identical to computing the shortest distance between a sphere and a spherical-cone.

This problem is solved in two steps,

1. Compute the point in the axis nearest to the center of the sphere.

$$\lambda_j^{\perp} = \frac{(P_i^* - P_{j0}) \cdot (P_{j1} - P_{j0})}{| P_{j1} - P_{j0} |^2} \tag{16}$$

2. Update the value of λ_j, to obey the angular condition represented by Equation 8.

$$\lambda_j = \lambda_j^{\perp} + \frac{| P_i^* - P_j(\lambda_j^{\perp}) |}{| P_{j1} - P_{j0} |} \cdot \tan \alpha_j \tag{17}$$

where $\alpha_j = \arcsin \eta_j$

Distance Between Volumes Generated from Dynamic Spheres with Two-degrees of Freedom

Distances between spherical-planes (two-degree of freedom) will involve determining the distances between spherical-vertices and spherical-planes as well as between spherical-edges. All possibilities are included by these two types of comparisons.

Spherical-edges are one-degree of freedom geometries, so they have been considered previously. However, given that an exhaustive distance computation between all the spherical-edges will be considered highly inefficient, several rules will have to be considered to reduce the number of elements potentially involved and therefore the number of comparisons.

The distance between a spherical-vertex or simply a sphere and a spherical-plane is computed by a process similar to the one for computing the distance between a point and a plane. That is, first by projecting the point onto the plane and then by measuring the distance between the point and the projected-point.

In our case, a sphere projected onto a spherical-plane produces a "projected-sphere" inside the spherical-plane in such a way that the distance between the two volumes is obtained by measuring the distance between these two spheres.

The process for computing the distance between spherical-vertices of volume i and the spherical-plane j is solved in two stages.

1. Compute the perpendicular to the surface of spherical-plane j.

   The unitary vector perpendicular to the surface, $\vec{n}_{js}$, must obey the following equations

$$\begin{aligned} \vec{n}_{js} \cdot \vec{v}_{j1} &= \eta_{j1} \\ \vec{n}_{js} \cdot \vec{v}_{j2} &= \eta_{j2} \end{aligned} \tag{18}$$

   where

$$\eta_{j1} = \frac{(R_{j1} - R_{j0})}{| P_{j1} - P_{j0} |}. \quad \eta_{j2} = \frac{(R_{j2} - R_{j0})}{| P_{j2} - P_{j0} |} \tag{19}$$

$$\vec{v}_{j1} = \frac{(P_{j1} - P_{j0})}{\mid P_{j1} - P_{j0} \mid}, \quad \vec{v}_{j2} = \frac{(P_{j2} - P_{j0})}{\mid P_{j2} - P_{j0} \mid} \tag{20}$$

A pseudo-spherical coordinates system based on vectors $\vec{v}_{j1}$, $\vec{v}_{j2}$ and $\vec{n}_{jc} = \vec{v}_{j1} \times \vec{v}_{j2}$ is introduced to compute the vector $\vec{n}_{js}$, according to the Equation 12. In this case the values of the basic variables are

$$\phi_j = \arctan(\frac{\eta_{j2} - \eta_{j1} \cdot \cos a}{\eta_{j1} \cdot \sin a}), \quad \theta_j = \arccos \frac{\eta_{j1}}{\cos \phi_j}, \quad \rho = 1 \tag{21}$$

where $a = \arccos(\vec{v}_{j1} \cdot \vec{v}_{j2})$

2. Project each spherical-vertex of volume i onto volume j and compute the distances:
   (a) Each spherical-vertex of volume i, $\{(P_{v_{ik}} . R_{v_{ik}}),\ k = 1, K\ ,\ K \geq 3\}$, is associated with a projected-sphere in the volume j, $(P_{pv_{ik}} . R_{pv_{ik}})$. The center of the projected sphere is computed as follows

$$P_{pv_{ik}} = P_{v_{ik}} + d_{ik} \cdot \vec{n}_{sj} \tag{22}$$

   where the distance between centers is computed by

$$d_{ik} = \frac{(P_{j0} - P_{v_{ik}}) \cdot \vec{n}_{jc}}{\cos \beta_j} \tag{23}$$
$$\cos \beta_j = \vec{n}_{jc} \cdot \vec{n}_{js}$$

   When the center, $P_{pv_{ik}}$, has been computed the radius, $R_{pv_{ik}}$, and the parameters $\lambda's$ for the projected-sphere need to be obtained. To compute λ_{j1} and λ_{j2}, multiply the first Equation 5 by $\vec{n}_{jc_2} = \vec{n}_{jc} \times \vec{v}_{j2}$ and $\vec{n}_{jc_1} = \vec{n}_{jc} \times \vec{v}_{j1}$ respectively, to give

$$\lambda_{j1} = \frac{(P_{pv_{ik}} - P_{j0}) \cdot \vec{n}_{jc_2}}{(P_{j1} - P_{j0}) \cdot \vec{n}_{jc_2}}$$

$$\lambda_{j2} = \frac{(P_{pv_{ik}} - P_{j0}) \cdot \vec{n}_{jc_1}}{(P_{j2} - P_{j0}) \cdot \vec{n}_{jc_1}} \tag{24}$$

   (b) The distance between the spherical-vertex and the spherical-plane, supposed it unbounded, is given by

$$D_{ik} = d_{ik} - R_i(\lambda_{i1}, \lambda_{i2})_{v_{ik}} - R_j(\lambda_{j1}, \lambda_{j2})_{pv_{ik}}, \quad \forall \lambda's \in \Re \tag{25}$$

If the spherical-vertex corresponding to the minimum distance, $(D_i^* = \min_k\{D_{ik}\})$, generates a projected-sphere inside the spherical-plane, $(\lambda's \in [0,1])$, then this minimum distance is actually the distance between the two objects. Otherwise, the values of the $\lambda's$ of the projected-sphere for all the spherical-vertex will determine up to four spherical-edge to spherical-edge distance computations.

Numerical Results and Example

A set of algorithms has been implemented in order to verify the theory and to determine the simplicity and efficiency of the technique presented in this paper. The spherical-geometries generated from dynamic-spheres have been used to represent various robotic systems and their environments.

The distance computation algorithms have been tested on a standard SUN SPARCstation 1, a RISC-based workstation, using sets of one hundred randomly generated and positioned volumes. The minimum, maximum, and average times in milliseconds for computing distance between all the combinations of objects developed in the theory have been recorded in the following table.

Objects Compared	Minimum Time (ms)	Maximum Time (ms)	Average Time (ms)
sphere to sphere	0.05	0.05	0.05
sphere to spherical-cylinder	0.14	0.14	0.14
sphere to spherical-cone	0.25	0.26	0.26
sphere to spherical-plane	0.65	0.91	0.87
spherical-cylinder to spherical-cylinder	0.20	0.23	0.23
spherical-cylinder to spherical-cone and spherical-cone to spherical-cone	0.65	1.57	1.05
spherical-cylinder to spherical-plane* and spherical-cone to spherical-plane*	0.72	3.86	2.82
spherical-plane* to spherical-plane*	2.81	9.09	7.01

(*) For these computations, planes with 4 vertices have been considered

A robotic testbed platform. used at the CIRSSE to study robotic assembly of structures in space, has been modeled in order to thoroughly test the entire theory. The setup consists of a PUMA 560 and a PUMA 600 each mounted on a three-degree of freedom platform. Each platform can be independently translated along a common track and has two rotational degrees of freedom, rotation about a vertical axis and tilt about a horizontal axis.

A kinematic model based on homogeneous transformation matrices using a modified form of the Denavitt-Hartenberg parameters [20] is used in order to determine the position of the robot links.

Each link of the robotic system has at least one associated volume as can be seen in Figure 5. In particular each individual PUMA robot-arm is modeled by the following spherical-volumes: A sphere and a spherical-cylinder for Link 0, a spherical-cylinder for Link 1, spherical-planes for links 2 and 3, this latter one including the wrist, and a sphere as an approximation to the gripper. A more precise representation of the latter objects could be constructed using several smaller volumes. An infinite plane is used to define the floor. The description of each volume in terms of its end-spheres with respect to its coordinate frame appears in the following table. The spherical-plane representing Link 2 has been clipped by two relations in the form of Equation 6 whose parameters do not appear in the table.

Name	Type	Coordinate Frame	X(mm)	Y(mm)	Z(mm)	R(mm)
Link 0	Sphere	P0	0	604	0	305
	cylinder	P0	0	0	0	83
			0	0	597	83
Link 1	Cylinder	P1	0	-108	0	83
			0	152	0	83
Link 2	Plane	P2	-146	-149	0	51
			-146	149	0	51
			470	149	0	51
Link 3	Plane	P3	86	76	0	43
			-86	76	0	43
			-86	-356	0	43
Wrist	Cylinder	P4	0	0	0	61
			0	0	-38	61
Hand	Sphere	P1a	0	-62	0	43

When considering self-collision for the PUMA robot arm, the set of volumes described above can be simplified to give even a better approximation. In terms of distance computation, links 2 and 3 are represented better by a combination of spherical-cones and spherical cylinders. This is due to the fact that certain links can never collide and in addition, certain surfaces will never be involved in the computation of shortest distances between elements in the same robot-arm. The following table shows the pairs of elements which need to be checked for self-collision.

	Link 0		Link1	Link 2	
	sphere	sph-cylinder	sph-cylinder	sph-cylinder	sph-cone
Link 3 and wrist spherical-cone	X	-	X	-	-
gripper sphere	X	X	X	X	-

As can be seen in the table, only six distances need to be computed: one sphere to sphere, three sphere to spherical-cylinder, one sphere to spherical-cone, and one spherical-cylinder to spherical-cone. Taking values from the table of computation times, the minimum, maximum, and average times in milliseconds are 1.37, 2.30, and 1.78 respectively. The maximum error in distances with respect to the exact shape of links 0 to 3 plus the sphere at the end is less than one centimeter. Collisions with the floor are determined immediately from the z-coordinates of the sphere bounding the gripper and one end-sphere of the spherical-cone for Link3.

For the overall robotic system, using the volumes shown in Figure 5, the number of comparisons required are 1 sphere to sphere, 14 sphere to sphere-cylinder, 11 sphere to spherical-plane, 20 spherical-cylinder to spherical-cylinder, 25 spherical-cylinder to spherical-plane and 12 spherical-plane to spherical-plane. This gives minimum, maximum and average times in milliseconds of 66, 220 and 170 respectively.

When restricting the volumes to zero or one degree of freedom, using spherical-cones for links 2 and 3 plus wrists, as well as replacing platforms with hemispheres, the computation time is reduced significantly. In this case the number of comparisons are as follows, 6 sphere to

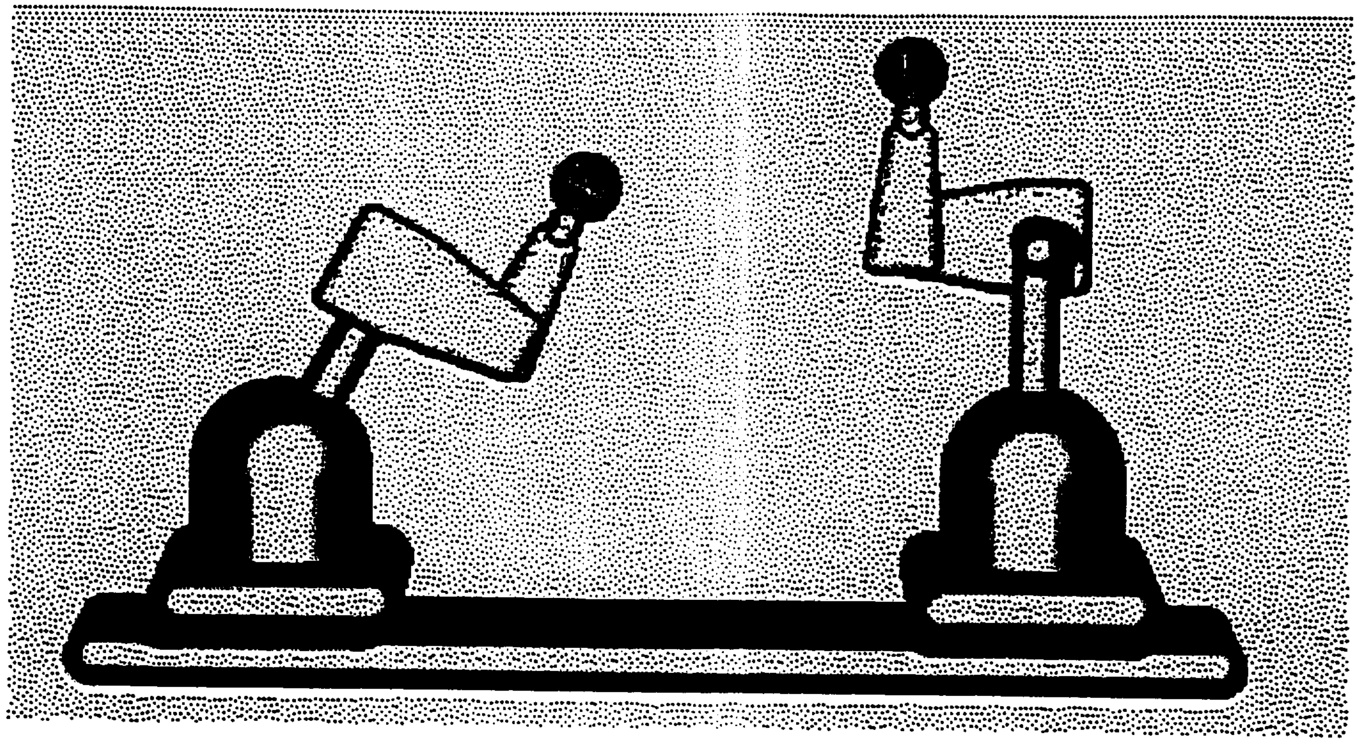

Figure 5: Robotic-system: Cooperative robot-arms

sphere, 14 sphere to sphere-cylinder, 14 sphere to sphere-cone, 10 spherical-cylinder to spherical-cylinder, 15 spherical-cylinder to spherical-cone and 4 spherical-cone to spherical-cone. This gives minimum, maximum and average times in milliseconds of 20, 38 and 28 respectively. Details of these numerical results are presented and discussed in a CIRSSE report[21].

Conclusions

A new three-dimensional object representation technique for generating spherical-volumes has been presented based on the idea that any real object can be approximated by an infinite number of spheres. These spheres are easily described by introducing the concept of dynamic-spheres. Although only linear functions have been used for describing the dynamic-spheres, the set of volumes generated is quite extensive and has been proved to be enough to model a robotic system with reasonable accuracy. The concept is easily extended to include other kinds of functions.

A very fast procedure for computing distances between spherical-volumes has been developed with good numerical results. The procedure has been shown to be simple and efficient when dealing with robotic systems.

Futher work will focus on using other kinds of functions for the dynamic-spheres, with the intent of: -increasing the variety of volumes generated, e.g., in two-degree of freedom a dynamic-sphere could follow general surfaces in order to give a better approximation of complex objects; -considering basic movements such as spheres rotating around an axis to describe the movement of a robot-arm in terms of the volumes swept by its links.

Acknowledgement

Josep Tornero gratefully acknowledges the support of this research through a grant from the Spanish Government (MEC-DGICYT). Gregory J. Hamlin and Robert B. Kelley appreciate the support in part by Grant NAGW-1333 from NASA to the Center for Intelligent Robotic Systems for Space Exploration.

References

[1] J. O'Rouke and N.I.Badler, "Decomposition of three-dimensional objects into spheres," *IEEE Trans. PAMI*, vol. 1, July 1979.

[2] B. Agin, *Representation and Description of Curved Objects.* PhD thesis, AIM-173, Stanford AI Laboratory, October 1972.

[3] S. Udupa, "Collision detection and avoidance in computer controlled manipulators," in *Proc. 5th Int. Conf. on Artificial Intelligence*, (MIT, Cambridge, Massachusetts), pp. 737–748, August 1977.

[4] T. Lozano-Perez and M. Wesley, "An algorithm for planning collision-free paths among polyhedral obstacles," *Commun. ACM*, vol. 22. pp. 560–570, October 1979.

[5] T. Lozano-Perez, "Automatic planning of manipulator transfer movements," *IEEE Trans. Systems Man and Cybernetics*, vol. SMC-11, pp. 681–698, October 1981.

[6] C. Chung and G. Saridis, "Path planning for an intelligent robot by the extended vgraph algorithm," in *IEEE International Symposium on Intelligent Control*, (Albany, New York), September 1989.

[7] C. Chung and G. Saridis, "The recursive compensation algorithm for avoidance path planning," in *IEEE Intrernational Workshops on Intelligent Robots and Systems*, (Tsukuba, Japan), September 1989.

[8] R. Brooks, "Solving the find-path problem by good representation of free space," *IEEE Trans. on Systems, Man and Cybernetics*, vol. SMC-13, pp. 190–197, March/April 1983.

[9] R. Brooks and T. Lozano-Perez, "A subdivision algorithm in configuration space for find-path with rotation," *IEEE Trans. on Systems. Man and Cybernetics*, vol. SMC-15, pp. 224–233, March/April 1985.

[10] V. Hayward, "Fast collision detection scheme by recursive decomposition of a manipulator workspace," in *IEEE Int. Conf. on Robotics and Automation*, (San Francisco, California), pp. 1044–1049, April 1986.

[11] B. Faverjon, "Object level programming of industrial robots," in *IEEE Int. Conf. on Robotics and Automation*, (San Francisco, California), pp. 1406–1412, 1986.

[12] E. Gilbert, D. Johnson, and S. Keerthi, "A fast procedure for computing the distance between complex objects in three-dimensional space," *IEEE Journal of Robotics and Automation*, vol. 4, no. 2, pp. 193–203. April 1988.

[13] J. Whitehead and K. Kyriakopoulos, "Efficient implementation of linear and quadratic programming algorithms for minimum distance estimation between solids," Tech. Rep.

CIRSSE-TR-89-22, Center for Intelligent Robotics Systems for Space Exploration, Rensselaer Polytechnic Institute, Troy, New York, 12180-3590, 1989.

[14] A. Sciomachen and P. Magnani, "A collision avoidance system for a space manipulator arm," in *Proceedings of the NASA Conference on Space Telerobotics*, vol. 5, pp. 283–291, 1989.

[15] D. Johnson and E. Gilbert, "Minimum time robot planning in the presence of obstacles," in *Proc. IEEE Conference on Decision and Control*, pp. 1748–1753, 1985.

[16] O. Khatib, "Real-time obstacle avoidanve for manipulators and mobile robots," *The Int. Journal of Robotics Research*, vol. 5, pp. 90–98. Spring 1986.

[17] M. Erdmann and T. Lozano-Perez, "On multiple moving objects," in *IEEE Int. Conf. on Robotics and Automation*, (San Francisco, California), pp. 1419–1424, February 1986.

[18] J. Canny, "Collision detection for moving polyhedra," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. PAMI-8, no. 2, pp. 200–209, 1986.

[19] E. Gilbert, "A new algorithm for detecting the collision of moving objects," in *IEEE Int. Conf. on Robotics and Automation*, pp. 8–14. 1989.

[20] J. Craig, *Introduction to Robotics. Mechanics and Control.* Addison-Wesley Publishing Company, 1986.

[21] J. Tornero, G. Hamlin, and R. Kelley, "Efficient distance functions using sphererical-objects and their application to the two-puma platform system," Tech. Rep. CIRSSE-TR-90-64, Center for Intelligent Robotics Systems for Space Exploration, Rensselaer Polytechnic Institute, Troy, New York, 12180-3590, 1990.